horizontal line

**IDC – Recommendation Systems**

Deep neural network for youtube video recommendation (2016)

**28 February 2023**

Submitters:

1. Harel Damari, ID: 305792020
2. Omer Lapidot, ID: 204710156

# Stage 1 – Anchor paper

# Intro – paper's objectives

The paper "Deep Neural Networks for YouTube Recommendations" presents a deep neural network-based recommendation system. The main objective of the paper is to research the effectiveness of DNN’s in recommendation systems (which was back at that time relatively new idea), especially for videos. The authors aim to solve the problem of providing personalized recommendations to users on the YouTube platform, taking into account both explicit and implicit features (time in video, age, gender, search terms). The proposed recommendation system outperforms the existing YouTube recommendation system based on matrix factorization, showing improved performance in terms of top-N accuracy and diversity of recommendations.

The paper discusses the challenges of developing a recommendation system for a platform like YouTube, which has a vast amount of user-generated content and diverse user preferences. The authors aim to create a system that can handle the sheer volume of data and provide users with personalized recommendations that are both relevant and diverse.

To address this challenge, the authors propose using deep neural networks (DNNs) as the model for the recommendation system. The input comes from multiple sources.

# Dataset selection

We choose the well know movie-lens 100k dataset to work with. Detailed EDA done on previous ex's.

In term of how to adapt the movie lens dataset to our mission, as the authors explained in the article, they actually were trying to solve a classification problem, where the movie with highest probability is the recommended movie – and one can take up to TOP-K movies.

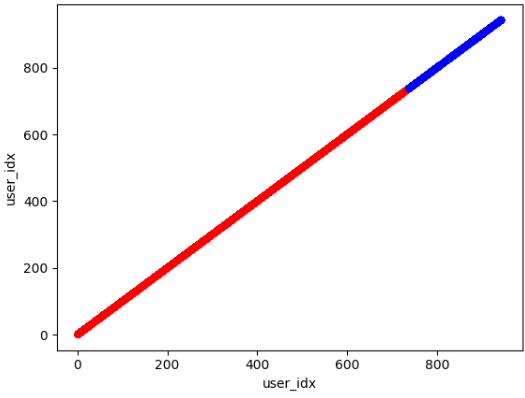
We adapt our dataset so we were building windows at length of 20, and the target movie is the movie that is 21 in the window, the window is sorted timestamp.

We then train the model to predict this target movie given the window along with more features as will be explained shortly, using Categorical-Cross-Entropy loss.

Building Train/Test sets:

As mentioned above we were building windows at the length of 20, and target, we also added user data. As we didn't want to blend to have data leak between the train-set and test-set we took 80% of users to train (all their windows) and test on the 20% of the rest.

If we would have shuffle, the model could exploit the fact that the windows are sorted with the timestamps – and "remember" the sequenced between encoded and coupled with the meta-data, so wanted strict separation between those windows.



# Algorithm architecture explanation

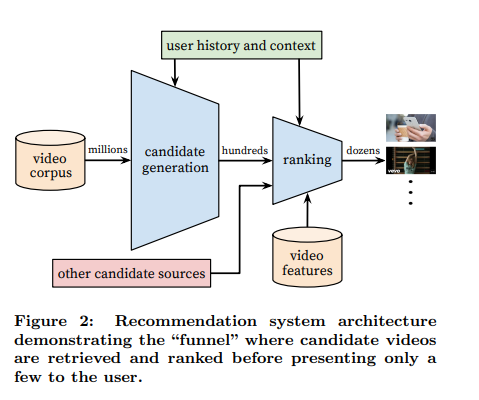
Overall architecture, the first mechanism is called “Candidate Generation” which considers data regarding the video’s, user history and context - and this module should filter from millions to hundreds of videos. Those hundred videos will be the input for the next module along with more user history and context, and more detailed features, this module will output the actual dozen recommendations for the user.

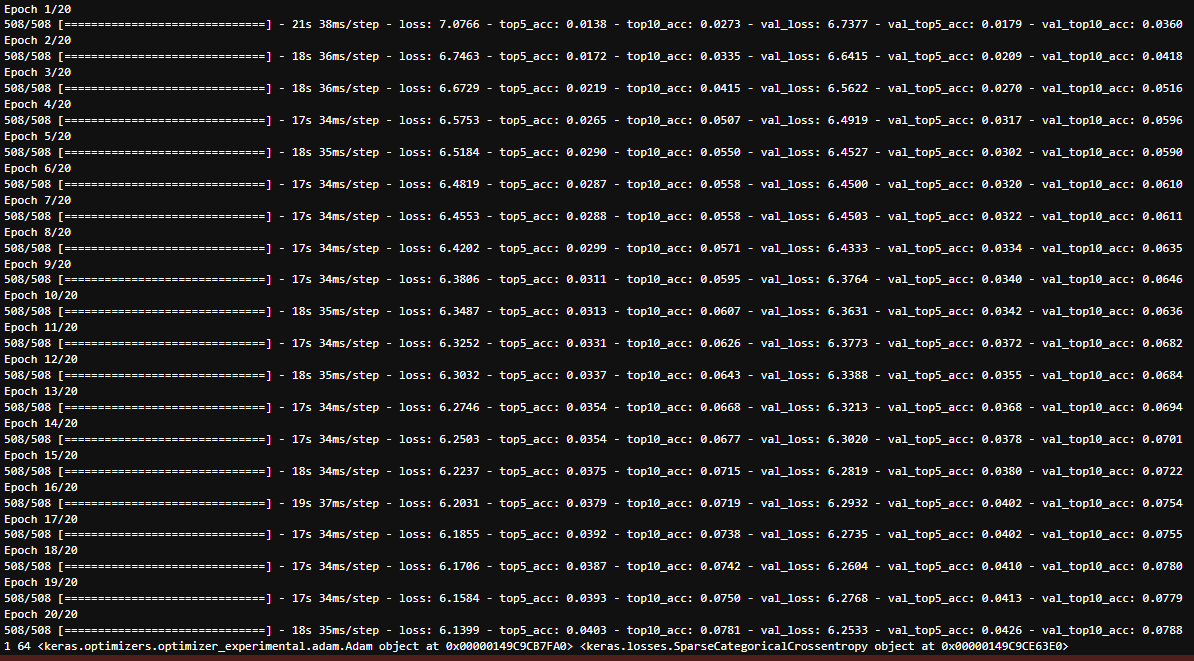
Model architecture: For both modules, at a glance, they take multiple input, concatenate them and feed to the MLP (multilayer-perceptron -> AKA fully connected), with Relu, the Non-linear activations.

One key thing is that the embeddings are learned along with all other model parameters. They also set as input, the user gender, logged-in state and age, along with search history.

תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטי





# 

# Stage 2 – improvement suggestion

# Algorithm architecture explanation

As in every ML system, there are 3 main parts that one can improve - We have the model, the training method, and the data. each one of these modules can be “fine tuned” independently.

We would like to suggest multiple ideas for each module, and we’ll implement an example from the suggestion we wrote.

Model:

* + - 1. Besides changing parameters regarding embedding size, dense size, activation functions or some other “fixed” numbers in the model architecture.
      2. Researchers can try different tensor-flow along the model in order to help the model extract the right gradients. This can be by adding an attention mechanism, trying to switch to CNN network, try one of the transformers architectures, try to use other pre-trained models in the model itself.

Training Procedure:

* + - 1. This includes adjusting the learning rate, using different optimizers, different number of epochs, and of course different loss.

Data:

* + - 1. You can extract more features from your data (connection between genres for example)
      2. Extend your dataset or use other dataset along with your dataset - for example, get the plot summary from google for each movie.
      3. Use synthetic data, which is not obvious when you want to mimic user actions.

**Our changes:**

* + - 1. (Small change) We added another data-layer - more data the model can exhaust we added an input (and then changed the architecture) of the mean genres of the input video.
      2. (Big change): In this change there’s different thinking of how to overtake the problem. Instead of feeding the network with a vector, we utilize the benefits of CNN, while keeping the same data structure (!), how?  
         Once we build our network input vector, we apply outer-product and feed forward to a convolutional neural network! (Which is completely different from the paper architecture - which uses only FC layer).
      3. An outer product is a mathematical operation that takes two vectors and produces a matrix as output. When the input vector is fed into a convolutional neural network (CNN) as an outer product, the resulting matrix can be treated as an image and processed by the network's convolutional layers - we treat the “recommendation context” as an image. This approach enhances the relationships between the elements in the vector and may be particularly useful when working with dense data, where it can allow the network to take advantage of spatial relationships between elements. It provides the network with additional information and structure, which lead to better performance on our task.

Code screenshots:

תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטי

# 

# We think those improvements will help the mode first, with ability to exploit the genres directly and not via embedding or search tokens, and second, with the ability to exploit connections between different elements in the input vector - for example: between age and genres, etc..

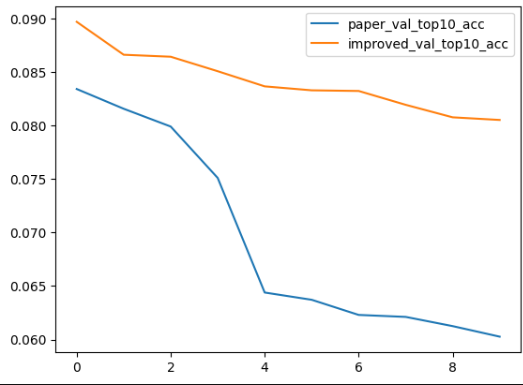
# 

# Performance metrics

Top-10 validation accuracy between paper implementation and our implementation.

This is the TOP-10 results from the grid search.

As can be seen, our models were superior over the paper model.



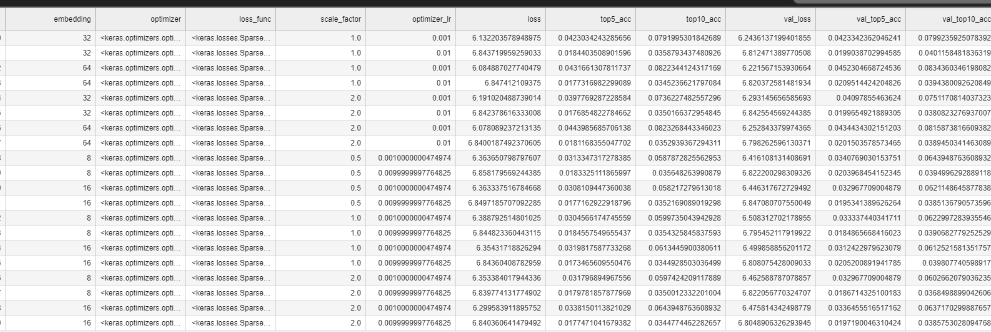
**Hyper Parameter optimization:**

All results are in the Repo under the csv files.

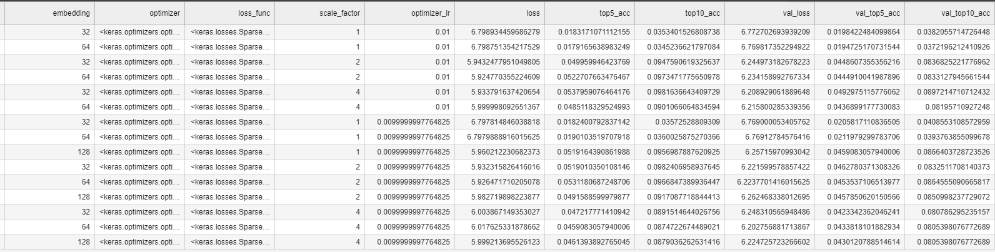
For the both the paper model and our model we had experiments with the:

* + - 1. Optimizer:
         1. Adam 0.01
         2. Adam 0.001
         3. SGD
      2. Embedding size:
         1. 8
         2. 16
         3. 32
         4. 64
      3. Loss:
         1. SparseCategoricalCrossentropy
      4. Scale factor:
         1. 0.5
         2. 1
         3. 2
         4. 4

**Paper Model:**

****

**Improvement Model:**

****

We ran brute force grid-search in order to find best parameters.

# Stage 3 – baseline algorithm

# Theory explanation

We compared a collaborative filtering algorithm based on KNN to our DNN.

# Performance metrics

Results show that TOP10 validation accuracy of 5.6% which is far lower than our models performance.

